INDEXING MATHEMATICAL SCHOLARLY PAPERS AS LINKED OPEN DATA

Azat Khasanshin
e-mail: azatkhasanshin@gmail.com
Danila Zaikin
e-mail: ksugltronteal@gmail.com
Nikita Zhiltsov
e-mail: nikita.zhiltsov@gmail.com

Kazan Federal University
18 Kremlyovskaya St.,
Kazan, Russia

Abstract

We present our work on developing an open source software platform for mining Linked Open Data (LOD) representation for a given collection of mathematical scholarly papers. Currently, the LOD cloud lacks up-to-date data on professional level mathematics. The main reason behind this is due to practical difficulties arising while dealing with such severe documents for indexing as mathematical papers that abound with formulas and specific structural elements ignored by the most state-of-the-art academic search engines. Our proof of concept demonstrates a feasible approach to parse these documents properly, dissect the semantics of their significant parts with the help of the ad hoc math-aware vocabulary, and publish their contents and metadata as RDF data. The authors argue that the platform at the final stage of its development cycle may be helpful for modern online scientific collections. For our experimental setup, we choose Math-Net.Ru – a digital collection well-known in the Russian mathematical community.

Keywords: *Indexing, Linked Open Data, Mining Logical Structure, Ontology Extraction*

1. INTRODUCTION

The Linked Open Data (LOD) initiative has recently emerged the added value of representing heterogeneous data from different content providers as a single interconnected "cloud" of objects[1]. The added value in such structured representation is in the standardized way of storing data integrated. As a rule, the data are loaded and transformed to RDF representation from such conventional data storage as relational databases, and, more rarely, from web pages or semi-structured textual documents using the so-called Linked Data principles, proposed by T. Berners-Lee [2]. Modern semantic search applications like a semantic search engine Sindice[2] or a mashup Sig.ma[3] harness the published RDF data to be able to either handle search queries more accurately or aggregate and display information about entities that users are interested in.

[1] http://lod-cloud.net
[2] http://sindice.com
[3] http://sig.ma

The data sets related to the domain of academic research papers form a significant and tightly connected subgraph in the LOD cloud. Namely, there are several LOD data sets affiliated with prominent digital collections e.g. ACM, DBLP, CiteSeer and others, that contain metadata of scholarly publications and links between each other. AKT Portal Ontology[4] is the de-facto standard for representing scholarly paper metadata in LOD. The schema covers a wide range of entities in the academic publishing domain including various types of publications (papers in conference proceedings, journal papers, technical reports etc.), author metadata (names, e-mail address, affiliation) and organizations. Among Semantic Web applications that exploit the data based on the AKT Portal Ontology, we would like to note RKBExplorer[5], which allows users to explore related authors, publications, organizations, and teaching courses.

However, the problem is that the academic data sets and their conjugate full text collections operate rather independently. Moreover, none of the data sets and collections contain information about particular entities, specific to some concrete subfield of science e.g. mathematics or physics. To our mind, eliciting and indexing the contents of such significant document parts as theorems, proofs, definitions, formulas etc. along with semantic relationships between them and, yet, paper metadata would allow obtaining a unified knowledge model for a given collection. Depending on the collection size, it will benefit professional researchers as well as learning students, by providing them more targeted services. The exemplary applications are article summarization and semantic search.

In the paper, we present our contribution in designing and implementing a programming solution to extract semantic LOD representation of mathematical scholarly papers in a given possibly connected digital collection.

2. RELATED WORK

«RDFizing» semi-structured or loosely structured documents is hot topic nowadays. Some tools primarily focus on conversion itself and do a little to add the semantics to data. For example, the Any23 library[6] accepts XHTML for converting web pages, possibly with RDFa annotations, to RDF. Recently, the library proves its reliability being used during large-scale extraction structured data from the Common Web Crawl [8].

[4] http://www.aktors.org/publications/ontology/

[5] www.rkbexplorer.com

[6] http://code.google.com/p/any23/

There are also numerous relevant works from the field of ontology extraction aka ontology learning that aim to analyze texts in terms of some domain-specific vocabulary, taxonomy or ontology. Impressive advances in ontology extraction have been achieved across many domains including bioinformatics [1], environment [7], law [6], and e-commerce [3].

To our knowledge, our work is the first attempt to extract RDF representation of indexed scholarly papers using not only their metadata, but also the text contents, in an automatic way.

3. ARCHITECTURE AND IMPLEMENTATION

Currently, our solution, which was implemented as a submodule of the Mocassin project[7], is characterized by the following main features:

- indexing mathematical papers in LaTeX as LOD compliant RDF data;
- mining the document logical structure using the math-aware Mocassin ontology[8];
- crawling Math-Net.Ru, one of the largest Russian online collections in mathematics, and ArXiv, for indexing publication metadata.

The overall infrastructure of the indexing process workflow is depicted in Figure 1. On the scheme, the ellipses represent various data sources including LaTeX source files, web resource API, web pages, vocabularies etc. The processing units are expressed as the bars. The RDF data storage is plotted as the cylinder. The following subsections give more details about the indexing process.

All the units were implemented mostly in Java, besides several third party utilities that were wrapped around by the ad hoc Java code. The platform also depends on the LaTeX distribution. We have used TeX Live[9] in our experiments.

3.1. Indexing mathematical papers in LaTeX

The main logic is implemented as a set of custom parsers for Apache Nutch, a widely used web crawler in Java. In particular, Nutch provides the basic infrastructure for retrieving LaTeX source files as indexing items and makes the indexing process either parallelizable (via conventional Java threads) or distributed (via Apache Hadoop). In general, it means that our approach is quite scalable with the evident bottleneck — the throughput performance of an underlying triple store. To achieve the best scalability, the triple store solution could be replaced by using HDFS files stored in the RDF/N3 format. In this case, the last reduce task may merge the triples from all the

[7] http://code.google.com/p/mocassin/
[8] http://cll.niimm.ksu.ru/ontologies/mocassin
[9] http://www.tug.org/texlive/

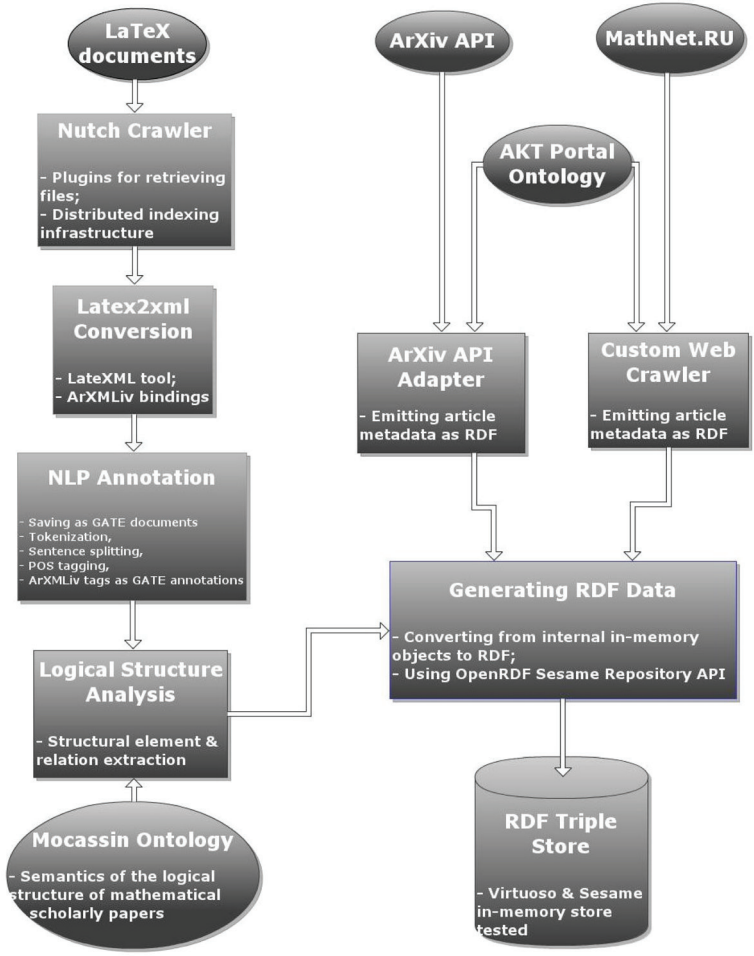

Figure 1. Indexing Process Workflow

nodes as preparation for saving data into the triple store without concurrent write operations. As we haven't yet tested our platform on the Hadoop cluster, we left this post-processing task out so far.

LaTeX is the input document format supported at the moment. The principal reason for that is two-fold, first, the need to handle the logical structure and formulas, and, second, its high popularity as an authoring format among mathematicians. However, operating programmatically with native

LaTeX is rather unhandy. That's why we use the ArXMLiv tools [5], which run on top of LaTeXML[10], a package of Perl scripts that do conduct LaTeX-to-XML conversion. The ArXMLiv distribution provides LaTeXML bindings (i.e. a mapping between standard LaTeX environments and XML elements) for wide range of available LaTeX packages and, therefore, gives opportunities to convert LaTeX source files into convenient XML representation.

3.2. NLP annotation

The next step of the indexing process is loading a given XML document into GATE[11], a platform for natural language processing. GATE recognizes elements from the ArXMLiv schema as GATE annotations, and executes in turn basic natural language processing tasks — tokenization, splitting sentences, and stemming. Thus, the output of the GATE processing resources per document is an in-memory Java object with mixed ArXMLiv and NLP annotations.

3.3. Mining the document logical structure

The mining procedure heavily uses our own ontology for modeling the semantic of certain document parts — Mocassin Ontology.

3.3.1. Mocassin Ontology

The ontology[12] aims to capture the semantics of the typical structural elements in mathematical scholarly papers. Each structural element represents the finest level of granularity and has its inherent features such as starting and ending positions, the text contents, and a functional role. In particular, it defines such ubiquitous document parts as theorems, lemmas, proofs , definitions, corollaries etc. Besides, the ontology asserts two types of object binary relations — navigational and restricted. The first relation type, which is represented by *refersTo* and *dependsOn* relations, tends to occur when the author points at significant parts of a publication in the form of referential sentences. The part-whole property (*hasPart*) and *followedBy* property belong to the first type too. An example of a relation of the second type is proves relation, which occurs between a proof — the only valid element type here — and a statement the proof justifies.

The ontology imports SALT Document Ontology[13] (SDO), an ontology of the rhetorical structure of scholarly publications. Specifically, it defines such classes as Section, Figure, and Table. In order to be able to make

[10] http://dlmf.nist.gov/LaTeXML/

[11] http://gate.ac.uk/

[12] The ontology's URL is http://cll.niimm.ksu.ru/ontologies/mocassin (login/password are demo/demokpfu)

[13] http://salt.semanticauthoring.org/ontologies/sdo

connections between structural elements and other objects contained by them and described elsewhere, e.g. mathematical named entities extracted from their text contents, we added a specific property — mentions — as follows: $mentions(x,y) \rightarrow (DocumentSegment(x) \vee Table(x) \vee Figure(x) \vee Section(x)) \wedge Thing(y)$. Document Segment is the root of the Mocassin ontology hierarchy. The ontology also defines classes to represent several types of mathematical expressions — Mathematical Expression, Variable, and Formula. The datatype property *hasLatexSource* is defined for storing LaTeX representation of the expression as a string.

In addition, the ontology contains a few cardinality axioms, e.g. one of them states that every proof must justify no more than one statements, and additional logical rules, e.g. $dependsOn(x,y) \wedge hasPart(z,y) \rightarrow dependsOn(x,z)$.

3.3.2. Ontology Extraction

Receiving the output from the previous step, this indexing unit mines the document logical structure by the method proposed in [4]. This procedure falls into two tasks: (i) recognizing the types of structural elements; (ii) recognizing the semantic relations between them. As a result, the module outputs a semantic graph that contains, on the one hand structural elements as nodes, each of which is assigned to a particular ontology class or marked "unrecognized" otherwise, and, on the other hand, ontology relation instances as edges. Aside from the functional properties, each node has the annotations corresponding to relevant datatype properties, i.e. title, the text contents, page numbers in the compiled PDF document.

3.4. Metadata extraction

3.4.1. Calling ArXiv API

ArXiv.org[14] is an open access digital collection of more than 750,000 scientific articles in physics, mathematics, computer science, biology, and economics. Along with access to human-oriented user interface, ArXiv provides machine access to its data including article metadata and LaTeX sources, if available, through ArXiv API[15]. Our platform is integrated with these facilities and is capable to handle the Atom/XML formatted response from ArXiv API.

We are going to install this functionality into the Nutch HTTP filters. At the moment, calling ArXiv API is executed as an independent task.

[14] http://arxiv.org
[15] http://arxiv.org/help/api

3.4.2. Crawling Math-Net.Ru

Math-Net.Ru[16] is an online digital collection of more than 100,000 scientific journal articles in mathematics. Math-Net.Ru provides free open access to the full texts in the PDF format after the period fixed per journal. Alas, the web resource does not provide machine access to paper metadata via API. However, that can be crawled due to the decent site structure and plain web page layouts. Every paper metadata can be accessed through a standardized URL e.g. «http://mathnet.ru/ivm18». The most bibliographical references were thoroughly extracted by the Math-Net.Ru content managers, and each reference is supplied with a hyperlink to the web page of the cited paper, if it is available on Math-Net.Ru.

We implemented a custom Math-Net.Ru crawler that is able to parse and collect available paper metadata in terms of the AKT Portal Ontology. Though, this is not integrated in the Nutch machinery yet. The reason is as follows. Math-Net.Ru does not provide access to LaTeX sources of the full texts, and we had to obtain them separately and use them locally, so it makes the indexing process non-smooth. We also discuss the copyright issues in Section 4.5.

3.5. RDF data generation

All the data from the previous steps flow together in this unit to be converted to RDF representation. For the purpose, we use the OpenRDF Sesame library written in Java, which prepares the RDF triple statements and saves them into the triple store — Virtuoso server instance. Virtuoso is a high-performance RDBMS server with extensive RDF/SPARQL support.

3.6. RDF data consumption: use cases

We have deployed our solution on the test bed and executed indexing available 1330 articles from the journal «Izvestiya Vuzov. Matematika». The indexing process took 5 hours on 8 core × 2.67GHz, 11GB RAM, SATA II Controller machine. The resulted RDF data can be accessed via SPARQL endpoint[17] in the form of the named RDF graph under the conditions provided in Section 4.5 of the paper. The data set contains over 290,000 RDF triples including the descriptions of over 2300 theorems, 1700 proofs, 1400 lemmas, 500 definitions and other mathematical entities indexed. We demonstrate several use cases as SPARQL queries below to illustrate possible applications.

[16] http://mathnet.ru

[17] SPARQL endpoint URL is http://cll.niimm.ksu.ru:8890/sparql-auth (login/password are demo/demokpfu)

Use case 1 (Search of theorems by keywords).

```
PREFIX m: <http://cll.niimm.ksu.ru/ontologies/
mocassin#>
SELECT ?theorem WHERE {
?theorem a m:Theorem;
m:hasText ?text .
?text bif:contains ''finite AND group''
}
```

In the header, the query declares the Mocassin ontology prefix. It also uses a class Theorem and a datatype property hasText with rdfs:Literal value from the ontology along with Virtuoso's specific property bif:contains for dealing with the triple store full-text index. As a result, the SPARQL processor will output the URIs of those theorems that contain a keyword «finite group» in their text bodies. One can get the related article URI through a part-whole like property hasSegment from the Mocassin ontology.

Use case 2 (Search of related theorems). Let's assume that two theorems from two different articles are semantically close or related, if either article cites the same third article.

```
PREFIX m: <http://cll.niimm.ksu.ru/ontologies/
mocassin#>
PREFIX akt: <http://www.aktors.org/ontology/portal#>
SELECT ?theorem1 ?theorem2 WHERE {
?paper1 m:hasSegment ?theorem1;
akt:cites-publication-reference ?paper3 .
?theorem1 a m:Theorem .
?paper2 m:hasSegment ?theorem2;
akt:cites-publication-reference ?paper3 .
?theorem2 a m:Theorem .
}
```

The query adds the AKT Portal Ontology prefix to the header and exploits cites-publication-reference property from this ontology and elements occured in the first query.

Use case 3 (Detection of the main results).

Using the query below, one could receive information about the structural graph per particular document in order to determine one or several important structural elements (theorems, proofs, corollaries, equations etc.) that may be the central results in the paper.

```
PREFIX m: <http://cll.niimm.ksu.ru/ontologies/
mocassin#>
SELECT ?from ?relation ?to WHERE {
```

```
<http://mathnet.ru/ivm18> m:hasSegment ?from .
<http://mathnet.ru/ivm18> m:hasSegment ?to .
?from ?relation ?to .
}
```

Out of this data one could form a structural graph with structural elements as nodes and relations between them as edges and apply any well-known importance metrics, e.g. eigenvalue centrality or PageRank scores. For instance, in case of using hasPart, refersTo, dependsOn and followedBy relations from the Mocassin ontology and PageRank, we can interpret elements with highest scores as the most probable elements for a reader to stay on after long sequence of transitions between them while looking through a document.

4. DISCUSSION, WORK IN PROGRESS AND FUTURE WORK

4.1. Mining the logical structure

We are working on improvement of the techniques of mining the logical structure. In particular, our current goal is to build a unified machine learning solution that incorporates ontology axioms, our knowledge about mathematicians' conventions for authoring and statistical evidences. We choose Markov Logic Networks as a convenient theoretical framework to achieve this goal and are going to add this novel mining approach to the platform in future releases.

4.2. Citation link analysis

Often citation contexts, i.e. anchors and anchor texts, in mathematical papers are of interest, because some of them may induce relations between structural elements from different publications. For example, there is a proof that contains a citation: «… we complete this proof by analogy to the proof of Theorem 1 from [Gabdulkhaev, 2004]». If these two documents were processed and are in our index, we could detect such new relations and add them to the index by analyzing occurrences of structural element names or abbreviations. Such a treatment of citations may be helpful in distinguishing «hard» links that matter during describing a proof and justifying some result from «soft» links that are used for referencing related works in the area.

4.3. Understanding formula symbol meanings

Many theoretical results' descriptions may contain no meaningful words at all and state some facts using only the elements of mathematical notation. These symbols can be defined either in the article earlier or in classical

textbooks in the field. Therefore, we consider a task of automatically understanding mathematical symbol meaning is of high priority.

4.4. Technical issues

Finally, we see some architectural discrepancies concerning two-way data processing for metadata and sources that need to be resolved. We also need optimizing our algorithms and data structures to speed-up the overall process. Yet, we have to test our solution in a multi-machine cluster and measure impact of performance improvement.

4.5. License issues and data ownership

As it has been mentioned previously by different authors in the field, publishing LOD inevitably faces the copyright issues, if the publishers and the data holders are not the same. To date, we have a written agreement with Kazan Federal University, the copyright holder of the «Izvestiya Vuzov. Matematika» journal to use a collection of the papers along with their LaTeX sources published in 1997-2009 for our research purposes. According to Math-Net.Ru's Terms of Use, «all materials published on this website including full-text articles, abstracts and author indexes are fully copyrighted by Steklov Mathematical Institute, Russian Academy of Sciences, and/or by other copyright holder, publisher, founder, editorial board …» and «reproduction or republication of the materials contained on Math-Net.Ru in any form requires written permission of the copyright holder». Relying on these principles, we consider eligible publishing the paper metadata and the data generated from the given «Izvestiya Vuzov. Matematika» collection as a copyright of Kazan Federal University. Thus, the content, which is accessible via the given SPARQL endpoint URL, may be copied, modified and redistributed only after gaining permission of the copyright holder.

5. CONCLUSION

We present a platform for mining structured standardized representation of scholarly papers in mathematics. It can be used for automatic publishing their contents as well as metadata in the format of LOD-compliant data. We apply the tool on a collection of over 1300 publications to demonstrate feasibility of the solution. We provide several use cases to illustrate utility of the published data.

6. ACKNOWLEDGMENTS

This work was supported by the Russian Foundation for Basic Research (grant 11-07-00507-a). The authors would like to thank the members of Computational Linguistics Laboratory at Kazan Federal University, especially professor Valery Solovyev and Olga Nevzorova, for their support and fruitful discussions.

REFERENCES

1. **Baker, C., Kanagasabai, R., Ang, W., Veeramani, A., Low, H., Wenk M.** 2007. Towards Ontology-driven navigation of the lipid biblioshere. Proceedings of the 6th International Conference on Bioinformatics.
2. **Berners-Lee, T.** Linked Data – Design Issues. – W3C. – 2006. - http://www.w3.org/DesignIssues/LinkedData.html
3. **Liu, W., Jin, W., Zhang, X.** Ontology-based User Modeling for E-commerce System // Proceedings of the 3rd International Conference on Pervasive Computing and Applications (ICPCA). – 2008.
4. **Solovyev, V., Zhiltsov, N.** Logical Structure Analysis of Scientific Publications in Mathematics // Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS'11).- ACM.- 2011.- P. 21:1-21:9.
5. **Stamerjohanns, H., Kohlhase, M., Ginev, D., David. C., and Miller, B.** 2010. Transforming Large Collections of Scientific Publications to XML. Mathematics in Computer Science, 3(3), 299-307.
6. **Volker, J. Fernandez-Langa, S., Sure, Y.** 2008. Supporting the Construction of Spanish Legal Ontologies with Text2onto // Computable Models of the Law. – Springer. – 2008.
7. **Volker, J., Haase, P., Hitzler, P.** Learning Expressive Ontologies // Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge. – IOS Press. – 2008. - P. 45-69.
8. Web Data Commons. Bizer, C., Muhleisen, H., Harth, A., Stadtmuller, S. http://webdatacommons.org/